

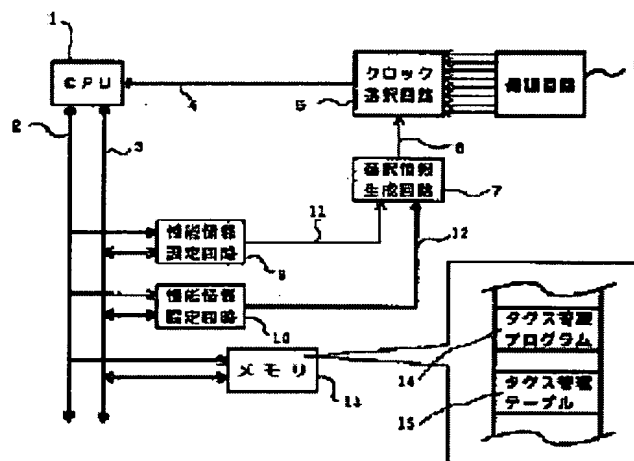
DEVICE AND METHOD FOR CONTROLLING CLOCK OF CENTRAL PROCESSING UNIT

Patent number: JP8076874
Publication date: 1996-03-22
Inventor: NISHIOKA KIYOKAZU; TANAKA KAZUHIKO; NOGUCHI YOSHIKI; OBA SHINYA
Applicant: HITACHI LTD
Classification:
 - international: G06F1/04; G06F1/06; G06F9/46
 - european:
Application number: JP19940212448 19940906
Priority number(s):

Abstract of JP8076874

PURPOSE: To control a clock of CPU so as to execute an operation with a low power consumption by automatically changing-over the clock into the absolute min. operation clock of CPU within a range where the requesting performance of a program to run is satisfied in the operation environment of a multi- task.

CONSTITUTION: The clock controlling device is provided with a performance information of a central processing unit(CPU) 1, which is required at every task, and also provided with more than one performance information setting circuits 9 and 10 setting performance information of CPU 1 at every task, a selecting information generating circuit 7 deciding the clock frequency of CPU 1 so as to permit the operation to be executed with absolute min. performance which is required by the task in starting, an oscillation circuit 6 generating plural clock signals and a clock selecting circuit 5 selecting one of the clock signals and giving it to the CPU 1.



Data supplied from the esp@cenet database - Worldwide

(19) 日本国特許庁 (J P)

(12) 公開特許公報 (A)

(11) 特許出願公開番号

特開平8-76874

(43) 公開日 平成8年(1996)3月22日

(51) Int.Cl. ⁸	識別記号	庁内整理番号	F I	技術表示箇所
G 0 6 F 1/04	3 0 1 C			
1/06				
9/46	3 4 0 Z	7737-5B		
			G 0 6 F 1/04	3 1 0 A
			審査請求	未請求 請求項の数7 O L (全 15 頁)

(21) 出願番号 特願平6-212448

(22) 出願日 平成6年(1994)9月6日

(71) 出願人 000005108

株式会社日立製作所

東京都千代田区神田駿河台四丁目6番地

(72) 発明者 西岡 清和

神奈川県川崎市麻生区王禅寺1099 株式会社日立製作所システム開発研究所内

(72) 発明者 田中 和彦

神奈川県川崎市麻生区王禅寺1099 株式会社日立製作所システム開発研究所内

(72) 発明者 野口 孝樹

東京都国分寺市東恋ヶ窪一丁目280番地
株式会社日立製作所中央研究所内

(74) 代理人 弁理士 高橋 明夫 (外1名)

最終頁に続く

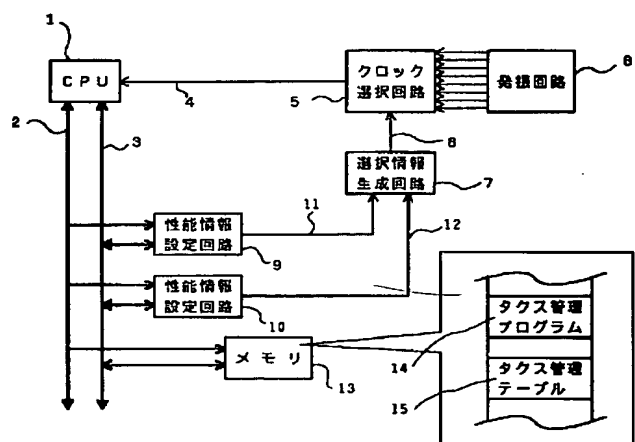
(54) 【発明の名称】 中央処理装置のクロック制御装置およびクロック制御方法

(57) 【要約】

【目的】 マルチタスクの動作環境において、走行させるプログラムの要求性能を満足する範囲において、必要最低限のCPUの動作クロックに自動的に切り替えて消費電力を節約し、低消費電力で作動するようにCPUのクロックを制御する。

【構成】 タスク毎に必要とする中央処理装置1の性能情報を設けて、前記タスク毎の中央処理装置1の性能情報を設定する一つ以上の性能情報設定回路9、10と、起動中のタスクが必要とする必要最低限の性能で動作するように前記中央処理装置1のクロック周波数を決定する選択情報生成回路7と、複数のクロック信号を発生する発振回路6と、そのクロック信号の中から一つを選択して前記中央処理装置1へ与えるクロック選択回路5を設ける。

図 1



【特許請求の範囲】

【請求項1】 複数のタスクを起動し切り換えて実行できるマルチタスクのオペレーティングシステムとプログラムとを格納するメモリと、
前記マルチタスクのオペレーティングシステム環境下で、前記プログラムを実行し、かつ、与えられるクロック周波数に基づいて動作スピードが決定される中央処理装置とを備える情報処理装置の中央処理装置のクロック制御装置において、
前記マルチタスクのオペレーティングシステム環境下で起動されるタスク毎に必要とする中央処理装置の性能情報を設けて、
前記タスク毎の中央処理装置の性能情報を設定する一つ以上の性能情報設定回路と、
前記性能情報設定回路に設定した一つ以上の性能情報を用いて、起動中のタスクが必要とする必要最低限の性能で動作するように前記中央処理装置のクロック周波数を決定する様に選択情報を生成する選択情報生成回路と、
複数のクロック信号を発生する発振回路と、
前記選択情報に応じて、前記複数のクロック信号の中から一つを選択して前記中央処理装置へ与えるクロック選択回路を設けたことを特徴とする中央処理装置のクロック制御装置。

【請求項2】 前記中央処理装置と、前記一つ以上の性能情報設定回路と、選択情報生成回路と、クロック選択回路を1チップ内に集積することを特徴とする請求項1記載の中央処理装置のクロック制御装置。

【請求項3】 複数のタスクを起動し切り換えて実行できるマルチタスクのオペレーティングシステムとプログラムとを格納するメモリと、
前記マルチタスクのオペレーティングシステム環境下で、前記プログラムを実行し、かつ、与えられるクロック周波数に基づいて動作スピードが決定される中央処理装置とを備える情報処理装置の中央処理装置のクロック制御方法において、
前記マルチタスクのオペレーティングシステム環境下で起動されるタスク毎に必要とする中央処理装置の性能情報を設けて、
一つ以上の性能情報設定回路と、
選択情報生成回路と、
複数のクロック信号を発生する発振回路と、
クロック選択回路とを有し、
前記一つ以上の性能情報設定回路が、前記タスク毎の中央処理装置の性能情報を設定し、
前記選択情報生成回路が、前記性能情報設定回路に設定した一つ以上の性能情報を用いて、起動中のタスクが必要とする必要最低限の性能で動作するように前記中央処理装置のクロック周波数を決定する様に選択情報を生成し、
前記クロック選択回路が、前記選択情報に応じて、前記

発振機から発生した複数のクロック信号の中から一つを選択して前記中央処理装置へ与えることを特徴とする中央処理装置のクロック制御方法。

【請求項4】 前記マルチタスクのオペレーティングシステムが、前記各タスクが必要とする性能情報を各タスク単位で管理し、
タスクを起動する際に、前記性能情報設定回路へ起動するタスクの性能情報を設定するステップを有することと、
さらに、タスクを終了する際に、前記性能情報設定回路に設定してある該当するタスクの性能情報を無効にするステップを有することとを特徴とする請求項3記載の中央処理装置のクロック制御方法。

【請求項5】 前記タスクを起動する際に、そのタスクの性能情報をタスク管理テーブルへ登録するステップと、起動中の全タスクの性能情報を読み出すステップと、前記全タスクの性能情報を用いて、必要最低限の中央処理装置の性能情報を算出するステップと、前記中央処理装置の性能情報を前記選択情報生成回路へ設定するステップを有すること、
さらに、タスクを終了する際に、そのタスクの性能情報をタスク管理テーブルから削除するステップと、起動中の全タスクの性能情報を読み出すステップと、前記全タスクの性能情報を用いて必要最低限の中央処理装置の性能情報を算出するステップと、前記中央処理装置の性能情報を前記選択情報生成回路へ設定するステップを有することを特徴とする請求項4記載の中央処理装置のクロック制御方法。

【請求項6】 前記プログラムに従って、前記中央処理装置が、前記性能情報設定回路から得る一つ以上の性能情報から、前記中央処理装置のクロック選択情報を生成することを特徴とする請求項1および請求項2記載のいずれかの中央処理装置のクロック制御装置。

【請求項7】 前記情報処理装置が、電源供給手段として、電池とAC電源のどちらでも使用でき、
電源供給手段の識別手段を設けることで、電池を使用しているときと、AC電源を使用しているときを識別し、
その識別した結果によって、前記中央処理装置が、前記性能情報設定回路から得る一つ以上の性能情報から、前記中央処理装置のクロック選択情報を生成することを特徴とする請求項6記載の中央処理装置のクロック制御装置。

【発明の詳細な説明】

【0001】

【産業上の利用分野】本発明は、中央処理装置のクロック制御装置およびクロック制御方法に係り、パーソナルコンピュータ（以下、単に「パソコン」という）や小形情報端末に代表される情報処理装置に使用される中央処理装置（Central Processing Unit、以下、「CPU」と略記する）の省電力化を達成するのに好適な中央処理

装置のクロック制御装置およびクロック制御方法に関する。

【0002】

【従来の技術】近年、電力需要の増加によって、社会全般にわたって電力の消費量が問題とされてきている。一方、このような状況下で、小形コンピュータ市場が急激に拡大し、パソコンの世界市場における設置台数は300万台を越えている。また、電池で使用する携帯型情報機器の要求もたかまりつつあり、そのような機器の低電力化も重要な課題になりつつある。

【0003】したがって、特に、このようなパソコンや携帯型情報機器などの情報処理装置の中核であるCPUの消費電力低減が注目されており、従来、多くの手法が試みられている。このようなCPUの消費電力低減手法の有力なものひとつとして、CPUのクロックを制御する手法がある。これは、特定の条件下において、低速なクロックでCPUを動作させることにより、CPUの消費電力の抑制を達成するものである。ここで、この特定の条件は、電源電圧が任意のレベルまで低下したことや、CPUに対する割り込みが発生したことなどがある。

【0004】さて、このようなCPUの消費電力低減手法に関して、特に、情報端末など通信機能を実現する上で要求が強いマルチタスク機能を有する情報処理装置に適用する発明としては、特開昭62-150416号公報に記載の「低消費電力状態への移行方式」がある。この発明は、複数のタスクを並行動作させるオペレーティングシステム(Operating System、以下、「OS」と略記する)を搭載したシステムにおいて、実行すべきタスクの有無を検出する手段と、コンピュータシステムを低消費電力状態にするの手段を設け、実行すべきタスクが無い場合にコンピュータシステムを低消費電力状態にする移行方式である。

【0005】

【発明が解決しようとする課題】上記従来技術は、コンピュータシステムを低消費電力状態に移行する方式について述べている。しかしながら、上記従来技術は、実行すべきタスクが存在するかどうかを判定し、実行すべきタスクがない場合のみ、低消費電力状態で動作するものであり、タスク実行中に低消費電力状態で動作しないため、その適用範囲が狭いという問題点があった。

【0006】また、近年は、多大な演算性能を要求するマルチメディア(動画、音声など)がシステムに取り込まれてくることが多くなっている。例えば、ワープロや表計算のソフトウェアと、テレビ会議のソフトウェアを一つのパソコンで動作させる場合が考えられる。ここで、前者のワープロや表計算のソフトウェアは、CPUが数10MIPS(百万命令/毎秒、Million Instruction Per Second)の性能ならば十分使いものになるが、後者テレビ会議のソフトウェアは動画及び音声の圧縮伸

張処理に加えて通信機能も必要になり、数100MIPSの性能を必要とする。一方、CPUの性能は、年率約1.6倍程度の急速な高性能化傾向にあり、数年で数100MIPSの性能に達するものと予測できる。しかしながら、数10MIPSと数100MIPSの動作状態における消費電力の差は非常に大きいので、数100MIPSの性能を持つCPUに数10MIPSのソフトウェアを動作させることは電力の無駄な消費である。

【0007】したがって、使用するソフトウェアに応じて性能を自動的に切り換える要請があったが、従来技術では、プログラムの動作スピードに応じて、CPUの性能を切り替えるという考え方はされていないという問題点があった。

【0008】本発明は、上記従来技術の問題点を解決するためになされたもので、その目的は、マルチタスクの動作環境において、その情報処理装置で動作させるプログラムの性能に応じて、低い性能で済む処理プログラムの実行時には、その要求性能を満足する必要最低限のCPUの動作クロックに自動的に切り替えて消費電力を節約して、タスク実行中でも低消費電力での作動を実現しうるCPUのクロック制御装置およびクロック制御方法を提供することである。

【0009】また、本発明の他の目的は、その情報処理装置がAC電源で作動するか電池で作動するかを判定し、電池で作動するときのみ、低消費電力で作動するCPUのクロック制御装置およびクロック制御方法を提供することである。

【0010】

【課題を解決するための手段】上記目的を達成するために、本発明の中央処理装置のクロック制御装置に係る発明の構成は、複数のタスクを起動し切り換えて実行できるマルチタスクのオペレーティングシステムとプログラムとを格納するメモリと、前記マルチタスクのオペレーティングシステム環境下で、前記プログラムを実行し、かつ、与えられるクロック周波数に基づいて動作スピードが決定される中央処理装置とを備える情報処理装置の中央処理装置のクロック制御装置において、前記マルチタスクのオペレーティングシステム環境下で起動されるタスク毎に必要なとする中央処理装置の性能情報を設けて、前記タスク毎の中央処理装置の性能情報を設定する一つ以上の性能情報設定回路と、前記性能情報設定回路に設定した一つ以上の性能情報を用いて、起動中のタスクが必要とする必要最低限の性能で動作するように前記中央処理装置のクロック周波数を決定する様に選択情報を生成する選択情報生成回路と、複数のクロック信号を発生する発振回路と、前記選択情報に応じて、前記複数のクロック信号の中から一つを選択して前記中央処理装置へ与えるクロック選択回路を設けたようにしたものである。

【0011】より詳しくは、上記中央処理装置のクロッ

ク制御装置において、前記中央処理装置と、前記一つ以上の性能情報設定回路と、選択情報生成回路と、クロック選択回路を1チップ内に集積するようにしたものである。

【0012】上記目的を達成するために、本発明の中央処理装置のクロック制御方法に係る発明の構成は、複数のタスクを起動し切り換えて実行できるマルチタスクのオペレーティングシステムとプログラムとを格納するメモリと、前記マルチタスクのオペレーティングシステム環境下で、前記プログラムを実行し、かつ、与えられるクロック周波数に基づいて動作スピードが決定される中央処理装置とを備える情報処理装置の中央処理装置のクロック制御方法において、前記マルチタスクのオペレーティングシステム環境下で起動されるタスク毎に必要とする中央処理装置の性能情報を設けて、一つ以上の性能情報設定回路と、選択情報生成回路と、複数のクロック信号を発生する発振回路と、クロック選択回路とを有し、前記一つ以上の性能情報設定回路が、前記タスク毎の中央処理装置の性能情報を設定し、前記選択情報生成回路が、前記性能情報設定回路に設定した一つ以上の性能情報を用いて、起動中のタスクが必要とする必要最低限の性能で動作するように前記中央処理装置のクロック周波数を決定する様に選択情報を生成し、前記クロック選択回路が、前記選択情報に応じて、前記発振機から発生した複数のクロック信号の中から一つを選択して前記中央処理装置へ与えるようにしたものである。

【0013】より詳しくは、上記中央処理装置のクロック制御方法において、前記マルチタスクのオペレーティングシステムが、前記各タスクが必要とする性能情報を各タスク単位で管理し、タスクを起動する際に、前記性能情報設定回路へ起動するタスクの性能情報を設定するステップを有することと、さらに、タスクを終了する際に、前記性能情報設定回路に設定してある該当するタスクの性能情報を無効にするステップを有するようにしたものである。

【0014】さらに詳しくは、上記中央処理装置のクロック制御方法において、前記タスクを起動する際に、そのタスクの性能情報をタスク管理テーブルへ登録するステップと、起動中の全タスクの性能情報を読み出すステップと、前記全タスクの性能情報を用いて、必要最低限の中央処理装置の性能情報を算出するステップと、前記中央処理装置の性能情報を前記選択情報生成回路へ設定するステップを有すること、さらに、タスクを終了する際に、そのタスクの性能情報をタスク管理テーブルから削除するステップと、起動中の全タスクの性能情報を読み出すステップと、前記全タスクの性能情報を用いて必要最低限の中央処理装置の性能情報を算出するステップと、前記中央処理装置の性能情報を前記選択情報生成回路へ設定するステップを有するようにしたものである。

【0015】また、上記目的を達成するために、本発明

の中央処理装置のクロック制御装置に係る発明の別の構成は、上記中央処理装置のクロック制御装置において、前記プログラムに従って、前記中央処理装置が、前記性能情報設定回路から得る一つ以上の性能情報から、前記中央処理装置のクロック選択情報を生成するようにしたものである。

【0016】さらに別の構成は、上記上記中央処理装置のクロック制御装置において、前記情報処理装置が、電源供給手段として、電池とAC電源のどちらでも使用でき、電源供給手段の識別手段を設けることで、電池を使用しているときと、AC電源を使用しているときを識別し、その識別した結果によって、前記中央処理装置が、前記性能情報設定回路から得る一つ以上の性能情報から、前記中央処理装置のクロック選択情報を生成するようにしたものである。

【0017】

【作用】本発明によれば、個々のプログラムの固有の性能情報を有し、マルチタスク化においてタスクの起動および終了時に、動作しているプログラムの性能情報によって選択情報生成回路で必要なCPUの性能を決定し、CPUのクロックを制御する。そのため、複数のプログラムが並列に実行している場合でも、各プログラムの要求性能を考慮してCPUの動作クロックを決定しているので、必要かつ最低の電力で動作可能になる。

【0018】また、電源検出回路によって、その情報処理装置の電源がAC電源か電池かを判定し、電源が電池のときには選択情報生成回路で、CPUのクロックを低い状態で作動させることにしておくことにより、電池で作動するときのみ、低消費電力で作動することになる。

【0019】

【実施例】以下、本発明に係る各実施例を、図1ないし図12を用いて説明する。

【実施例1】以下、本発明に係る第一の実施例を、図1ないし図6を用いて説明する。先ず、図1を用いて本発明に係るCPUのクロック制御装置の回路構成について説明しよう。図1は、本発明の第一の実施例に係るCPUのクロック制御装置の回路構成を示すブロック図である。

【0020】図1において、発振回路6は、8種類の周波数のクロックを生成することができる。選択情報回路7は、クロック選択回路5を制御するための回路である。性能情報設定回路9、10は、タスクの負荷情報を保持するための回路である。メモリ13には、CPU1が使用するプログラムやデータが格納される。そして、図に示される様に、タスク管理プログラム14およびタスク管理テーブル15は、このメモリ13に格納されるものである。

【0021】CPU1へは、5～40MIPSまでMIPS間隔で8レベルの動作状態を指定できるものとす

る。発振回路6は、上記各レベルに対応する8種類の周波数のクロックを出力する。

【0022】また、本発明に係る情報処理装置は、マルチタスクのOS (Operating System)が起動され、そのOS上で各種プログラムを実行することを前提としている。OSは、タスクの起動と終了を制御するタスク管理プログラム14を含んでいる。さらに、タスク管理プログラム14の制御に必要な各種情報がタスク管理テーブル15に含まれている。具体的には、各タスクごとに、各タスクをユニークに識別するタスクID、必要なメモリ容量などに加えて、本発明に特徴的な性能情報とペンディングフラグが登録される項目として存在している。

【0023】性能情報は、各タスクが実行するプログラムの内容に応じて指定されるCPUの負荷となる能力である。ペンディングフラグは、あるタスクの性能情報が性能情報設定回路に設定するための待ち状態になっていることを示すフラグである。

【0024】ペンディングフラグの役割は、後にフローを用いて本発明の動作を説明するときに明らかになるので、先に、性能情報について説明する。性能情報は、上で述べたように、タスクが実行するプログラムの内容に応じて指定する。例えば、ワードプロセッサに関するプログラムの場合で、編集プログラムは、10MIPS、印刷プログラムは、5MIPSのCPU性能が必要であるとする。

【0025】このような状況において、タスク管理プログラム14が、編集プログラムのタスクを起動する際に、性能情報設定回路9へ10MIPSの性能情報を設定すると、選択情報回路7は、10MIPSに相当するCPU1の動作周波数に対応する選択情報を生成する。この選択情報にしたがって、クロック選択回路5は該当するクロックを選択して、クロック線4を介して、CPU1へ供給する。これにより、CPU1は10MIPSで動作する。これが、本発明の基本的な仕組みである。

【0026】続いて、ワードプロセッサで編集した文書を印刷する際には、タスク管理プログラム14が印刷プログラムのタスクを起動する。このとき、性能情報設定回路10へ5MIPSの性能情報を設定する。選択情報回路7は、性能情報設定回路9に設定した10MIPSの性能情報と性能情報設定回路10に設定した性能情報から、15MIPSに相当するCPU1の動作周波数に対応する選択情報を生成する。この選択情報にしたがって、クロック選択回路5は該当するクロックを選択して、クロック線4を介して、CPU1へ供給する。これにより、CPU1は10MIPSから15MIPSへ、より高速な動作モードへ移行する。

【0027】性能情報は、時間ごとの処理能力であらわされるので、両タスクが並列実行されるばあいには、両タスクの加算になることに留意しよう。

【0028】本発明によれば、このように、ワードプロ

セッサで編集中に印刷を実行した場合でも、負荷に見合ったようにCPU1の性能を向上させて、印刷しながらでも編集のための十分な操作環境を得ることができるのである。

【0029】次に、図2を用いて性能情報設定回路の詳細な回路構成について説明しよう。図2は、本発明の第一の実施例に係る性能情報設定回路の回路構成を示すブロック図である。

【0030】上述したように、性能情報設定回路9は、タスク管理プログラム14によって起動される各タスクごとの性能情報を保持するための回路である。タスク起動時に、性能情報が性能情報設定レジスタ22へ設定されると共に、そのタスクのタスクIDがタスクID設定レジスタ20へ、さらに、タスクID設定レジスタ20と性能情報設定レジスタ22の設定が有効であることを示すイネーブル情報がイネーブル設定レジスタ21へ設定される。イネーブル設定レジスタ21は、論理値

「1」の時に有効状態、論理値「0」の時に無効状態を示す。したがって、論理積回路23は、有効状態の場合のみ、性能情報設定レジスタ22の情報を性能情報線11へ出力する。逆に、無効状態の場合、論理値「0」が性能情報線11へ出力される。

【0031】ここで、イネーブル設定レジスタ21が、有効状態のときに、この性能情報設定回路に設定された値が有効であることを示し、逆の無効状態が、この性能情報設定回路に設定された値が無効であり、いわば、値が設定されていないことを示している。

【0032】また、タスクID設定レジスタ20、イネーブル設定レジスタ21および性能情報設定レジスタ22の設定情報は、アドレスバス2およびデータバス3を介して、CPU1が読みだし可能であり、タスク管理プログラム14が性能情報設定回路9の設定情報を知ることができる。

【0033】次に、図3を用いて選択情報生成回路の詳細な回路構成について説明しよう。図3は、本発明の第一の実施例に係る選択情報生成回路の回路構成を示すブロック図である。

【0034】本実施例は、CPU1へ8レベルの動作状態を指定できたことを想起しよう。したがって、この選択情報生成回路7は、3ビット情報を扱うものになる。

【0035】図3において、加算回路30は、3ビットの情報を加算する。キャリー信号線31は、加算結果が桁上がりした時に論理値「1」を示す。デコーダ回路35は、3ビットである。

【0036】加算回路30は、性能情報線11の性能情報(0~7)と性能情報線12の性能情報(0~7)の加算結果を、論理和回路32~論理和回路34へ出力する。加算結果が7以下の場合、キャリー信号線31が論理値「0」となり、論理和回路32~論理和回路34は、加算回路30の加算結果をそのままデコーダ回路3

5へ出力する。

【0037】一方、加算回路30の加算結果が8以上の場合、加算結果7がデコード回路35へ出力される。要するに、キャリア信号線31が論理値「1」となり、論理和回路32～論理和回路34は全て論理値「1」をデコード回路35へ出力する。このような場合分けは、クロック選択回路5が選択できるクロックが8種類に限られているために、性能情報の最高のレベルを7に押さえるため必要となるものである。例えば、クロック選択回路5が16種類のクロックを選択可能ならば、この選択情報回路の構成も異なったものになる。

【0038】このように論理和回路32～論理和回路34が出力する性能情報を受けて、デコード回路35は、該当するクロック信号を選択するための情報を選択情報信号線8へ出力する。

【0039】以上説明したように、選択情報回路7は、性能情報線11および12の情報から生成したクロック選択情報を、選択情報信号線8へ出力し、クロック選択回路5に適切なクロックを選択させるものである。

【0040】次に、図4および図5を用いてタスク管理プログラム14の詳細を説明しよう。図4は、タスク起動時のタスク管理プログラムの動作をあらわすフローチャートである。図5は、タスク終了時のタスク管理プログラムの動作をあらわすフローチャートである。

【0041】まず、図4を用いてタスク管理プログラム14がタスクを起動する場合の動作を、図の順を追って説明しよう。最初に、通常のマルチタスクOSがおこなう所定のタスク起動処理がおこなわれる（S400）。ここでは、タスク管理テーブル15の内容が更新される。すなわち、タスク管理テーブル15に、新しく起動するタスクのタスクID、必要なメモリ容量、性能情報などが登録される。

【0042】次に、性能情報設定回路9、10に設定されているイネーブル情報を読み込み（S401）、その情報が有効か無効かをチェックする（S402）。これは、使われていない性能情報設定回路があるか調べるものである。

【0043】どちらかのイネーブル情報が無効状態ならば、該当する方の性能情報設定回路（9、10のどちらか）へ、起動するタスクのタスクID、イネーブル有効情報及び性能情報を設定する（S403）。これにより、新しいタスクが起動された環境下において最適なクロック周波数でCPU1が動作することになり、タスク起動処理を終了する。

【0044】一方、S402のステップにおいて、イネーブルが全て有効状態ならば、性能情報設定回路は、すべて他のタスクに使用されている状態である。この場合には、新しく起動したタスクによって、これらの性能情報設定回路9、10の性能情報を更新するべきかどうかを調べる必要がある。

【0045】この場合は、これらの性能情報設定回路9、10の性能情報を読み込む（S404）。

【0046】次に、読み込んだ各々の性能情報と、起動するタスクの性能情報を、比較する（S405）。その結果、起動するタスクの性能情報が最も低い値であるならば、既に起動されているタスクによって、十分にCPUの性能が上昇している状態である。よってこの場合は、性能情報設定回路9、10の更新せずに、タスク管理テーブル15中の起動するタスクのペンディングフラグを有効状態にして（S407）、終了する。ここで、ペンディングフラグは、タスクとしては、起動したものの、そのことが性能情報設定回路9、10には、影響を与えていないことを示している。このペンディングフラグは、後のタスク終了時に参照される。

【0047】一方、起動するタスクの性能情報が最も低い値でないならば、この起動されたタスクに対応して、CPUの性能を上昇させなければならない。したがって、この場合には、最も低い値が設定されている方の性能情報設定回路（9、10のどちらか）へ、起動するタスクのタスクID、イネーブル有効情報及び性能情報を設定する（S406）。そして、その書換えた性能情報設定回路のプロセスIDに対応するプロセスのペンディングフラグを有効にして、終了する。これは、書換えにより、既に起動されているタスクが、性能情報設定回路の性能に反映されなくなったため、後で復帰させるためである。

【0048】これにより、新しいタスクが起動された環境下において最適なクロック周波数でCPU1が動作することになり、タスク起動処理を終了する。

【0049】以上説明したように、本実施例は、2個の性能情報設定回路9と10で構成しているが、この個数を増やす場合にも対応可能であり、個数を増やすほどきめ細かなクロック制御が可能になり、省電力化の効果を得ることができる。

【0050】まず、図5を用いてタスク管理プログラム14がタスクを終了させる場合の動作を、図の順を追って説明しよう。最初に、性能情報設定回路9および10に設定されているタスクID情報を読み込み（S501）、その情報が終了しようとするタスクのタスクIDと一致しているかをチェックする（S502）。これは、終了しようとするタスクの性能が性能情報設定回路に反映されているか調べるためのものである。

【0051】どちらかの性能情報設定回路に格納されたタスクIDと終了しようとするタスクのタスクID情報が一致するならば、該当する方の性能情報設定回路（9、10のどちらか）へ、イネーブル無効情報を設定する（S503）。これは、性能情報設定回路の情報を消去したことに該当する。

【0052】次に、タスク管理テーブル15中にペンディングフラグが有効状態となっているタスクが存在する

かをチェックする(S504)。ペンディングフラグが有効状態となっているということは、そのタスクが起動されており、性能情報設定回路9、10に性能を設定するために待ち状態になっていると考えることができる。

したがって、ペンディングフラグが有効状態となっているタスクが存在するならば、そのタスクのタスクID、イネーブル有効情報及び性能情報を、ステップ503において該当した性能情報設定回路(9、10のどちらか)へ設定する(S505)。そして、設定したタスクのペンディングフラグを無効状態にする(S506)。

【0053】これにより、タスクが終了した環境下において最適なクロック周波数でCPU1が動作することになる。

【0054】最後に、タスク管理テーブル15から終了するタスクに関する情報を削除するなど、通常のマルチタスクOSが行う所定のタスク終了処理がおこなわれ(S507)、タスク終了処理を終了する。

【0055】一方、ステップS502において、どちらかのタスクID情報も一致しないならば、終了するタスクはCPU1の動作周波数決定に影響していないことになる。したがって、ステップS503～S506を飛び越して、ステップS507を実行し、タスク終了処理を終了する。

【0056】同様に、ステップS504において、ペンディングフラグが有効状態のタスクが存在しないならば、CPU1の動作周波数決定に影響するべきタスクがないことになる。したがって、ステップS505、S506を飛び越して、ステップS507を実行し、タスク終了処理を終了する。

【0057】最後に、図6を用いて以上説明した実施例の具体的な動作の例を経時順に説明してみよう。図6は、各タスクの状態と性能情報の関係を経時順に示したタイミングチャートである。

【0058】より詳しくは、図6は、各タスクの起動と終了を示すイベントと、起動されたタスクの状態(実行状態と待機状態間の遷移)と、性能情報設定回路9および性能情報設定回路10に設定される性能情報と、CPU1の実動作に相当する性能情報を示したものである。この図における時間軸の単位として、各イベントを起点とした6つのタイムスロット(0～5)を用いることにした。

【0059】図6の様に、タスクA、タスクB、タスクCが起動されるのであるが、これらの各々に対応する性能情報は、「2」、「4」、「5」とすることにしよう。

【0060】まず、タイムスロット0では、OSだけが動作しており、実行中のタスクは存在しない。このとき、性能情報設定回路9および性能情報設定回路10にはイネーブルが無効状態に設定されており、CPU1

は、最低性能「0」で動作している。

【0061】次に、タスクAが起動されたタイムスロット1では、タスク管理プログラム14が性能情報設定回路9へイネーブル有効情報と性能情報「2」を設定する。これにより、CPU1は性能情報「2」に相当する性能で動作することになる。

【0062】次に、タスクBが起動されたタイムスロット2では、まず、タスクBが実行状態となり、タスクAは実行状態から待機状態へ遷移する。さらに、タスク管理プログラム14が性能情報設定回路10へイネーブル有効情報と性能情報「4」を設定する。これにより、性能情報設定回路9に設定された性能情報「2」と性能情報設定回路10に設定された性能情報「4」が、選択情報生成回路7で加算されて、CPU1は性能情報「6」に相当する性能で動作する。この動作環境下で、タスクAとタスクBは、背反的に実行状態と待機状態の間を遷移する。

【0063】次に、タスクCが起動されたタイムスロット3では、まず、タスクCが実行状態となり、タスクAは実行状態から待機状態へ遷移する。起動されるタスクCの性能情報は、「5」なので、タスク管理プログラム14は、性能情報設定回路9および性能情報設定回路10へ設定されている性能情報を比較し、低い性能情報が設定されている方の性能情報設定回路9へ、性能情報「5」を設定する。これにより、本来性能情報は「9」となるが、本実施例では、性能情報の最大レベルが「7」なので、CPU1は性能情報「7」に相当する性能で動作する。

【0064】また、タスク管理プログラム14は、タスク管理テーブル15中のタスクAのペンディングフラグを有効状態とする。このような動作環境下で、タスクA、タスクB、タスクCは、背反的に実行状態と待機状態の間を遷移する。

【0065】次に、タスクBが終了されたタイムスロット4では、まず、タスクCが実行状態となり、タスクBは終了する。さらに、タスク管理プログラム14は、タスクBの性能情報「4」の代わりにペンディングフラグが有効状態となっているタスクAの性能情報「2」を性能情報設定回路10へ設定する。すなわち、追い出されていたタスクAの性能情報を復帰させるわけである。これにより、CPU1は性能情報「7」に相当する性能で動作する。この動作環境下で、タスクAとタスクCは、背反的に実行状態と待機状態の間を遷移する。

【0066】最後に、タスクAが終了されたタイムスロット5では、まず、タスクCが実行状態となり、タスクAは終了する。さらに、タスク管理プログラム14は、終了するタスクAの性能情報「2」を無効にするため性能情報設定回路10へイネーブル無効情報を設定する。これにより、性能情報設定回路9だけが有効となり、CPU1は性能情報「5」に相当する性能で動作する。

【0067】以上説明した第一の実施例では、性能情報設定回路が2個の構成であるが、これに限定したわけではなく、性能情報設定回路の数を増やせば、CPU1の動作速度を、さらに、きめ細かく制御できる。同様に、CPU1のクロック周波数も8種類に限定したわけではなく、周波数の選択しを増やせば、さらにきめ細かな省電力制御が可能になる。

【0068】〔実施例2〕以下、本発明に係る第二の実施例を、図7を用いて説明する。図7は、本発明の第二の実施例に係るCPUのクロック制御装置の回路構成を示すブロック図である。

【0069】この第二の実施例は、基本的な構成と動作およびその思想は、同様のものによるものであるが、その特徴は、回路構成を集積して1チップ化するところにある。

【0070】発振器62は、クロック信号を発生し、分周回路61は、周波数が異なる8種類のクロック信号を発生する。また、低電力対応CPU60は、CPU1と、性能情報設定回路9、10と、選択情報生成回路7と、クロック選択回路5と、分周回路61とを1チップに集積化したプロセッサである。

【0071】この実施例では、性能情報設定回路9などの回路部が低電力対応CPU60に集積化されておるため、回路構成全体の部品点数を削減できるという利点がある。

【0072】また、低電力対応CPU60が分周回路61を内蔵するため、発振器62からのクロック信号線は1本で済むことになる。低電力対応CPU60に内蔵する回路は、第一の実施例で説明したように、比較的簡単なハードウェアで構成できるため、回路規模が比較的小さく、ピン数もCPU1と比較して、それほど増加することはないので、十分に集積回路として構成することは可能である。

【0073】この第二の実施例の効果としては、回路を集積化することによる小形化、低電力化だけでなく、ハードウェアの実装設計が容易になることもある。すなわち、クロック選択回路5をチップ内部に取り込んだことで、CPUクロックの高速化に伴い問題となるクロック信号の反射や干渉など実装上の技術課題が緩和されるのである。

【0074】〔実施例3〕以下、本発明に係る第三の実施例を、図8ないし図10を用いて説明する。図8は、本発明の第三の実施例に係るCPUのクロック制御装置の回路構成を示すブロック図である。

【0075】この第三の実施例の特徴は、第一の実施例において、性能情報設定回路9、10と選択情報設定回路7が生成する選択情報を、図8に示されるタスク管理プログラム71に従って、選択情報設定回路70が生成することにある。

【0076】ここで、選択情報設定回路70は、クロッ

ク選択回路5へ与える選択情報を設定する回路であり、タスク管理プログラム71は、選択情報を生成する機能を持つプログラムである。

【0077】以下、この選択情報設定回路70の機能を、図9を用いて詳細に説明しよう。図9は、本発明の第三の実施例に係る選択情報設定回路の回路構成を示すブロック図である。

【0078】本実施例でも、取扱える性能レベルは、8段階を想定しており、したがって、レジスタ80は、3ビットである。選択情報設定回路70においては、アドレスバス2とデータバス3を介して、CPU1が3ビットの性能情報を設定するとともに、設定した性能情報を読み出すことができる。設定する性能情報は、実施例1とは異なり、性能情報設定回路が生成するのではなく、タスク管理プログラム71によって生成する。この性能情報は、選択情報設定回路70の中のデコーダ回路35へ送られ、デコーダ回路35は、上記性能情報からクロック選択情報を生成して、選択情報信号線8へ出力し、これによって、最終的にCPU1の動作周波数が決められることになる。

【0079】次に、実施例1との相違も考慮に入れて、タスク管理プログラム71の性能情報を設定する動作の詳細を、図10の順を追って説明しよう。図10は、タスク管理プログラムの性能情報を設定する動作をあらわすフローチャートである。

【0080】まず、タスク管理プログラム71は、これからおこなう処理がタスクの起動処理か終了処理かをチェックする(S900)。

【0081】起動処理ならば、図4のステップS400と同様に、所定のタスク生成処理を行う(S901)。ここでは、起動するタスクに対応する各種情報をタスク管理テーブル15へ登録する。

【0082】逆に、終了処理ならば、図5のステップS507と同様に、所定のタスク終了処理を行う(S902)。ここでは、終了するタスクに対応する各種情報をタスク管理テーブル15から削除する。

【0083】ステップS901およびS902の次には、起動中のタスクが存在するかをチェックする(S903)。このチェックは、タスク管理テーブル15への登録の有無を調べれば良い。

【0084】登録があるならば、登録されている全タスクの性能情報を読み出す(S904)。次に、読み出した性能情報の総和を求めて、CPU1の性能情報を生成し(S905)、CPU1の性能情報の値が「8」を越えているかをチェックする(S906)。越えていないならば、そのCPU1の性能情報をレジスタ80へ設定する(S907)。

【0085】逆に、CPU1の性能情報の値が「8」を越えているならば、CPU1の性能情報の値を「7」(本実施例における性能情報の最大値)として(S90

8)、レジスタ80へ設定する(S907)。

【0086】また、ステップS903において、起動中のタスクが存在しないならば、CPU1の性能情報の値を「0」(本実施例における性能情報の最小値)として(S909)、レジスタ80へ設定する(S907)。

【0087】このように、タスクの起動時および終了時において、性能情報からCPU1へ与える選択情報を生成する機能をタスク管理プログラム71に持たせれば、この既脳をソフトウェアで実現できる。したがって、本実施例においては、実施例1とは異なり性能情報設定回路9、10が不要であり、ハードウェアの部品点数を削減できるという利点がある。

【0088】〔実施例4〕以下、本発明に係る第四の実施例を、図11および図12を用いて説明する。本実施例の特徴は、使用している電源に従って、CPU1の性能を制御することにある。

【0089】それを考慮して、図11を用いて、本実施例に係るCPUのクロック制御装置の回路構成と電源回路の構成について説明しよう。図11は、本発明の第四の実施例に係るCPUのクロック制御装置の回路構成と電源回路の構成を示すブロック図である。

【0090】選択情報生成回路100は、性能情報からクロック信号の選択情報を生成することができる。電源検出回路110は、電源供給手段を識別するための回路である。また、電源制御回路111は、電源供給手段をAC電池113にするか電池114にするかを制御する。電源供給手段の選択回路112は、実際にどちらの電源を採用するか切り替える回路である。ここで、AC電源113は、このCPUを用いた情報処理装置の外部から電力供給され、電池114は、情報処理装置に内蔵されることを想定している。これらAC電源113と電池114は、どちらか一方が電源供給手段として使用されるものであり、どちらを使用するかは、電源制御回路111によって決定される。電源制御回路111の指示

により、選択回路112はAC電源113と電池114のいずれかを選択して情報処理装置で電力を供給するものである。

【0091】また、電源検出回路110は、電源制御回路111が選択回路112へ指示した情報がCPU1によって読み出されることを可能にし、CPU1が現在どちらの電源供給手段を使用中であることを検出できる。

【0092】選択情報生成回路100がCPU1の性能情報を選択するのは、CPU1が情報を設定するアドレスバス2とデータバス3を介して入力されるデータにより決定される。

【0093】以下、図12を用いて、この選択情報回路100の構成と動作について詳細に説明しよう。図12は、本発明の第四の実施例に係る選択情報回路の回路構成を示すブロック図である。

【0094】2ポートRAM101は、アドレス4ビット、データ3ビット、デコーダ回路102は3ビットとして構成されている。なお、本実施例も性能レベルは、「0」から「7」までの、8段階を想定している。2ポートRAM101は、アドレスバス2およびデータバス3から情報が設定される。この設定された情報にしたがって、性能情報線11、12の4ビットの性能情報がアドレス情報となり、その結果として読み出された3ビットのデータがデコーダ回路102へ送られる。送られたデータから、デコーダ回路102は、8ビットのクロック選択情報を出力し、8種類クロック信号のうち1つを選択する。

【0095】ここで、次の表1と表2を用いて、この2ポートRAM101へ設定される情報と出力の例を説明しよう。表1は、AC電源113を使用している通常動作モード時の性能情報の設定を対照した表である。

【0096】

【表1】

表 1

アドレス (性能情報)				データ (設定情報)			
信号線11		信号線12		クロックレベル	信号線 8		
1	1	1	1	7	1	1	1
1	1	1	0	6	1	1	0
1	1	0	1	5	1	0	1
1	1	0	0	4	1	0	0
1	0	1	1	5	1	0	1
1	0	1	0	4	1	0	0
1	0	0	1	3	0	1	1
1	0	0	0	2	0	1	0
0	1	1	1	4	1	0	0
0	1	1	0	3	0	1	1
0	1	0	1	2	0	1	0
0	1	0	0	1	0	0	1
0	0	1	1	3	0	1	1
0	0	1	0	2	0	1	0
0	0	0	1	1	0	0	1
0	0	0	0	0	0	0	0

【0097】表2は、電池114を使用している省電力動作モード時の設定対照した表である。

【0098】

【表2】

表 2

アドレス (性能情報)				データ (設定情報)			
信号線11		信号線12		クロックレベル	信号線 8		
1	1	1	1	5	1	0	1
1	1	1	0	5	1	0	1
1	1	0	1	5	1	0	1
1	1	0	0	5	1	0	1
1	0	1	1	4	1	0	0
1	0	1	0	5	1	0	1
1	0	0	1	4	1	0	0
1	0	0	0	3	0	1	1
0	1	1	1	2	0	1	0
0	1	1	0	4	1	0	0
0	1	0	1	3	0	1	1
0	1	0	0	2	0	1	0
0	0	1	1	1	0	0	1
0	0	1	0	3	0	1	1
0	0	0	1	2	0	1	0
0	0	0	0	1	0	0	1
0	0	0	0	0	0	0	0

【0099】表1に示される性能情報の設定は、基本的に第一の実施例と同様のアルゴリズムによるものであり、性能情報線11と12の情報を加算するアルゴリズムである。したがって、実行しているタスクの性能情報の和が実際のCPU1のクロックレベルとして用いられる。

【0100】一方、表2に示される性能情報の設定は、電池駆動で使用している場合に用いられる場合のもので、CPU1がフルパワーで動作しないようなアルゴリズムである。こりアルゴリズムでは、CPU1が高速で動作するクロックレベル6と7の設定を使用しない、つまり、性能情報線11と12の総和が6以上の時はクロ

ックレベル5に設定するように工夫されている。

【0101】このように、第四の実施例では、電源供給手段の使用状態に応じて、CPU1へ与えるクロック周波数を制御することで、電池114を用いている場合は、処理性能は低下するものの、使用している電池の動作時間を長くできる効果がある。

【0102】

【発明の効果】本発明によれば、マルチタスクの動作環境において、その情報処理装置で動作させるプログラムの性能に応じて、低い性能で済む処理プログラムの実行時には、その要求性能を満足する必要最低限のCPUの動作クロックに自動的に切り替えて消費電力を節約し

て、タスク実行中でも低消費電力での作動を実現しうるCPUのクロック制御装置およびクロック制御方法を提供することができる。

【0103】また、本発明によれば、その情報処理装置がAC電源で作動するか電池で作動するかを判定し、電池で作動するときのみ、低消費電力で作動するCPUのクロック制御装置およびクロック制御方法を提供することができる。

【図面の簡単な説明】

【図1】本発明の第一の実施例に係るCPUのクロック制御装置の回路構成を示すブロック図である。

【図2】本発明の第一の実施例に係る性能情報設定回路の回路構成を示すブロック図である。

【図3】本発明の第一の実施例に係る選択情報生成回路の回路構成を示すブロック図である。

【図4】タスク起動時のタスク管理プログラムの動作をあらわすフローチャートである。

【図5】タスク終了時のタスク管理プログラムの動作をあらわすフローチャートである。

【図6】各タスクの状態と性能情報の関係を経時順に示したタイミングチャートである。

【図7】本発明の第二の実施例に係るCPUのクロック制御装置の回路構成を示すブロック図である。

【図8】本発明の第三の実施例に係るCPUのクロック制御装置の回路構成を示すブロック図である。

【図9】本発明の第三の実施例に係る選択情報設定回路の回路構成を示すブロック図である。

【図10】タスク管理プログラムの性能情報を設定する動作をあらわすフローチャートである。

【図11】本発明の第四の実施例に係るCPUのクロック制御装置の回路構成と電源回路の構成を示すブロック図である。

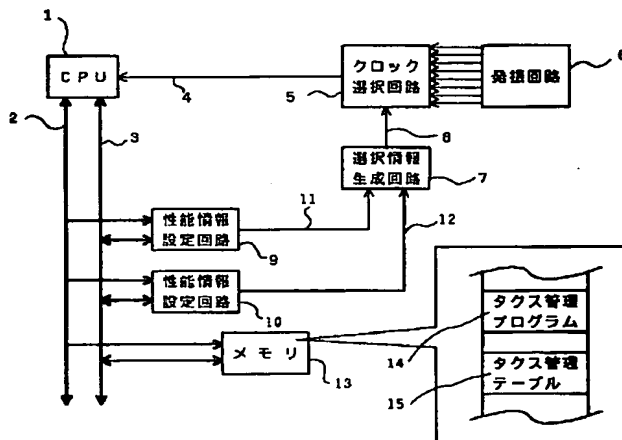
【図12】本発明の第四の実施例に係る選択情報回路の回路構成を示すブロック図である。

【符号の説明】

1…CPU、2…アドレスバス、3…データバス、4…CPU1のクロック線、5…クロック選択回路、6…発振回路、7…選択情報生成回路、8…選択情報信号線、9、10…性能情報設定回路、11、12…性能情報線、13…メモリ、14…タスク管理プログラム、15…タスク管理テーブル。20…タスクID設定レジスタ、21…イネーブル設定レジスタ、22…性能情報設定レジスタ、23…論理積回路。30…加算回路、31…キャリー信号線、32～34…論理和回路、35…デコード回路。60…CPU、61…分周回路、62…発振器。70…選択情報設定回路、71…タスク管理プログラム。80…レジスタ。100…選択情報生成回路、110…電源検出回路、111…電源制御回路、112…選択回路、113…AC電源、114…電池。101…2ポートRAM、102…デコード回路。

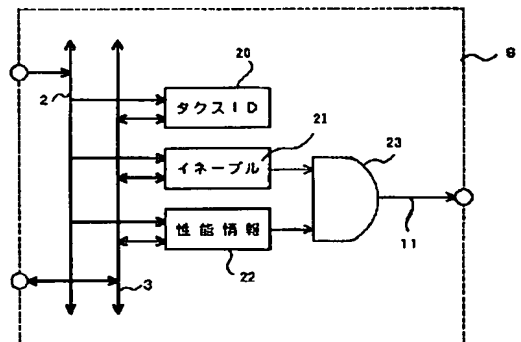
【図1】

図 1

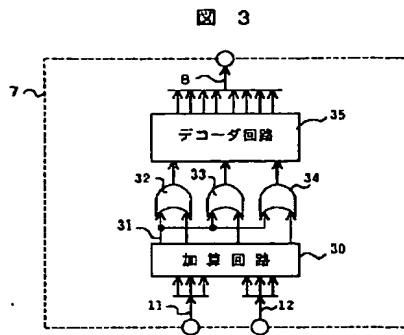


【図2】

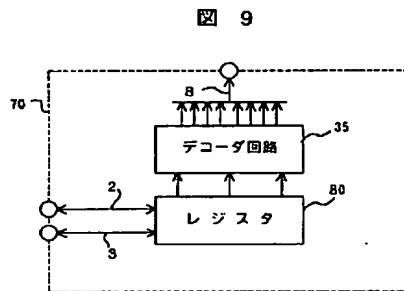
図 2



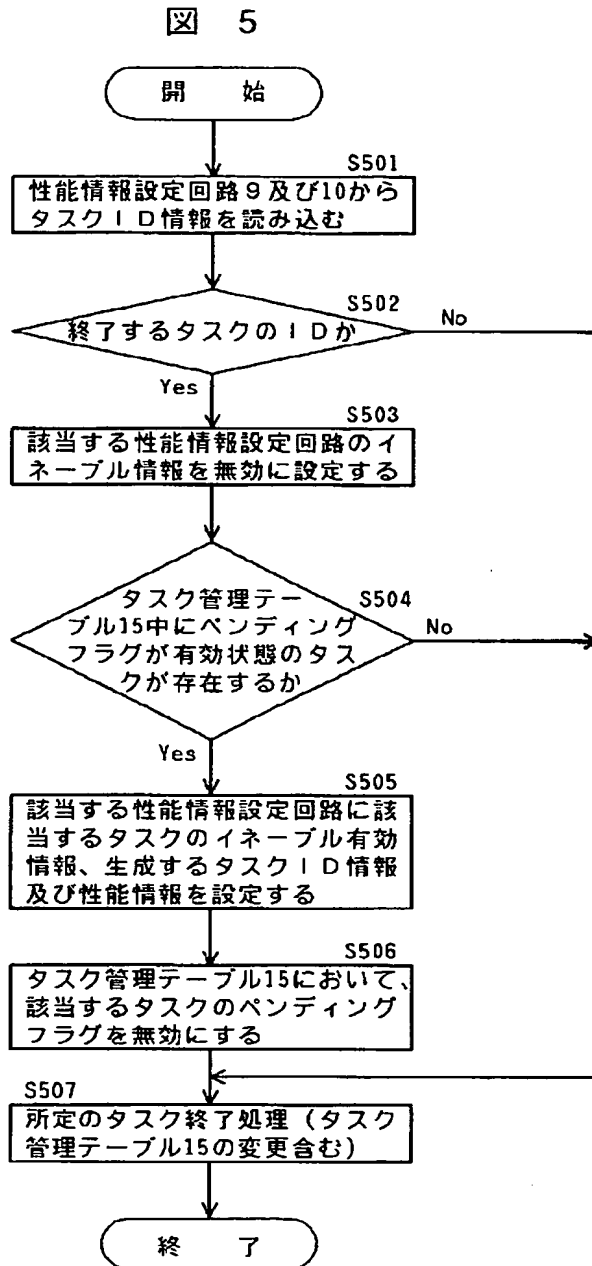
【図3】



【図9】

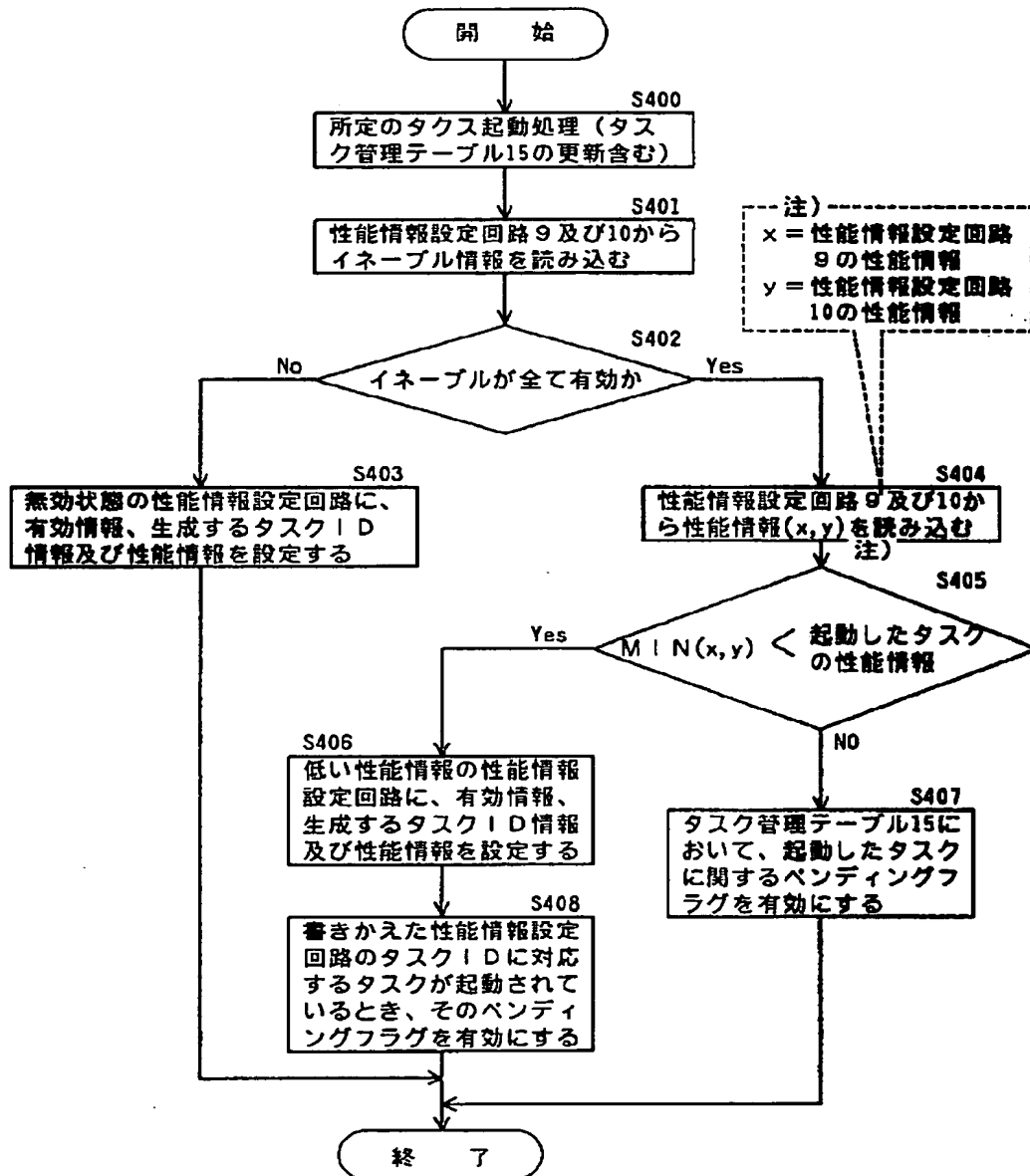


【図5】



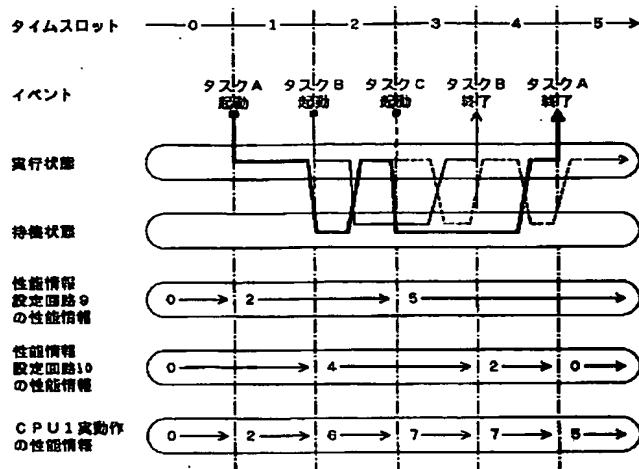
【図4】

図 4



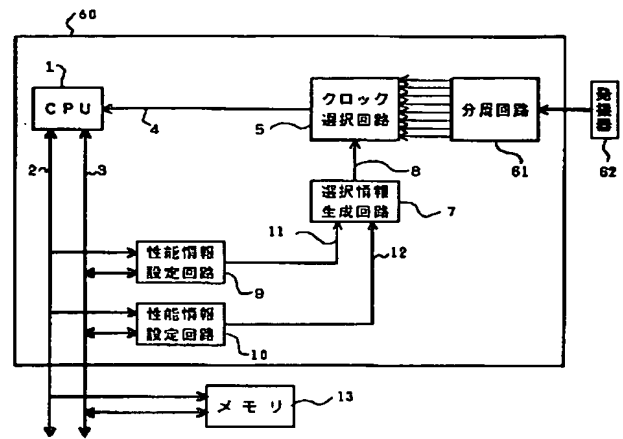
【図6】

図 6



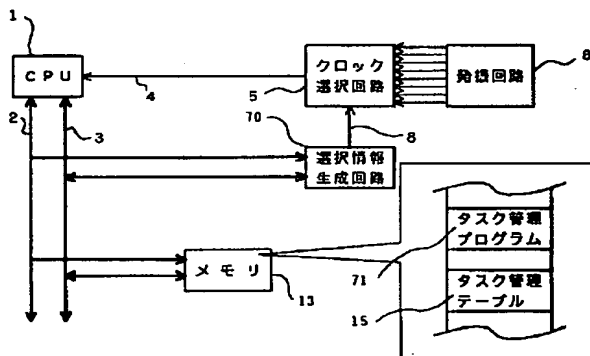
【図7】

図 7



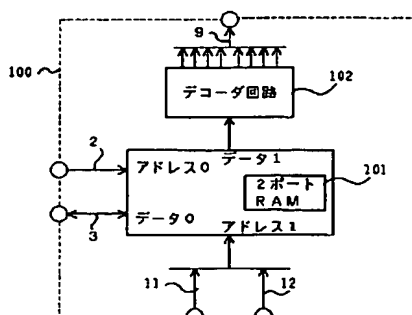
【図8】

図 8



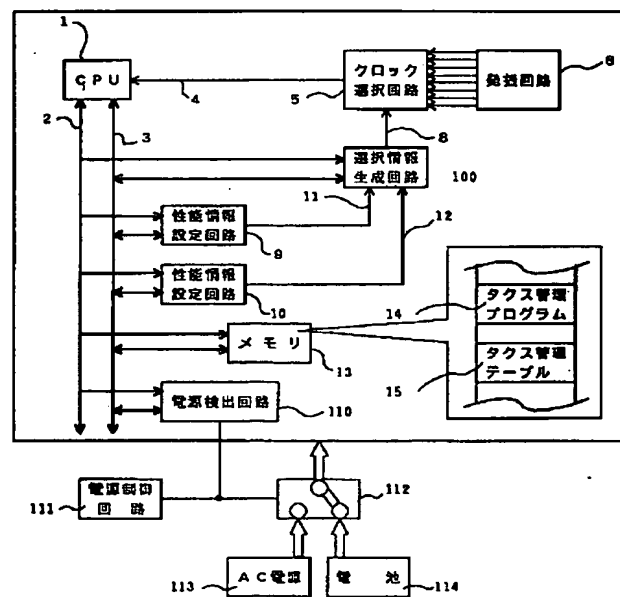
【図12】

図 12



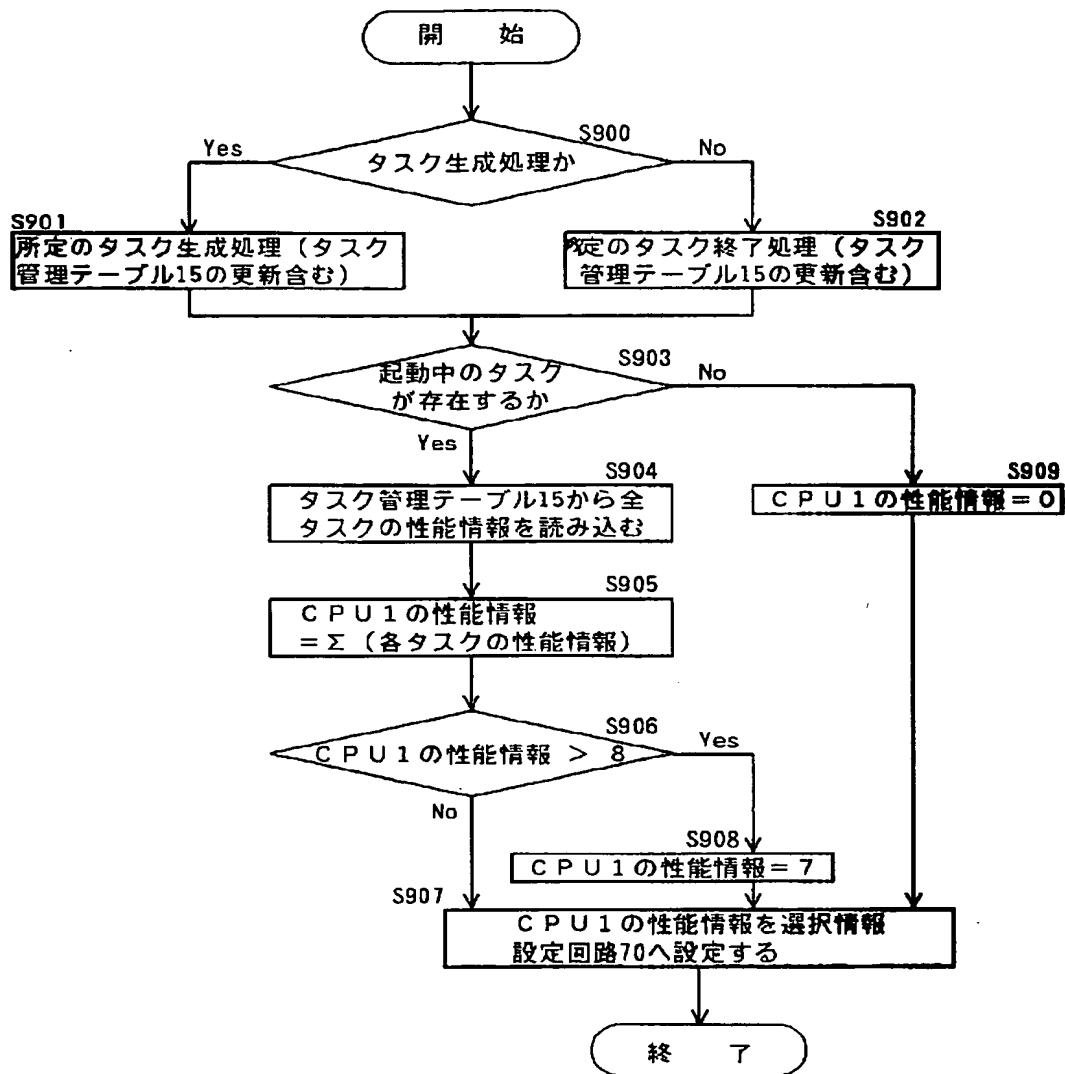
【図11】

図 11



【図10】

図 10



フロントページの続き

(72)発明者 大場 信弥
東京都小平市上水本町五丁目20番1号 株
式会社日立製作所半導体事業部内